(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 580 281 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **93303820.0**

(51) Int. Cl.5: **H04L 12/56**

(22) Date of filing: **18.05.93**

(30) Priority: **23.06.92 GB 9213240**

(43) Date of publication of application:
26.01.94 Bulletin 94/04

(84) Designated Contracting States:
DE FR GB

(71) Applicant: **International Business Machines Corporation**
Old Orchard Road
Armonk, N.Y. 10504(US)

(72) Inventor: **Judd, Ian David**
33 Coles Mead,
Otterbourne
Winchester, Hampshire SO21 2EG(GB)
Inventor: **Beer, Reginald**
4 Clevelands Close
Chandlers Ford, Hampshire SO5 1PX(GB)

(74) Representative: **Burt, Roger James, Dr.**
IBM United Kingdom Limited
Intellectual Property Department
Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) Network addressing.

(57) Described is a network addressing scheme in which a message sent from a source node to a destination node include a path address which defines the path over which the message should travel to reach the destination node. At each node between the source and destination, the path address is compared against a predetermined value, and on determining that the address and predetermined value are different, the node modifies the address before forwarding the message onto the next node. In a switch node having three or more ports, the identity of the output port is determined from the path address and a portion of the address is deleted before sending the message out on that output port. Also described is a method of configuring a network in which one or more initiator nodes are defined, the initiator nodes issuing query messages to an adjacent node which responds by sending the initiator details of the number of operational ports which are implemented in the adjacent node. The initiator node then issues query messages addressed to those nodes which are attached to the operational ports on the adjacent node. This continues until the initiator has walked through the entire network at which point the configuration process is complete.

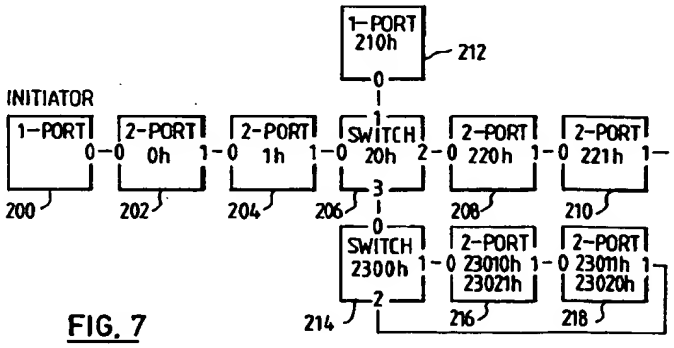

FIG. 7

EP 0 580 281 A2

This invention relates to network addressing and more particularly to a method of and apparatus for routing a message in a network of interconnected nodes.

A variety of different types of network configurations have been proposed or used for transmitting data between interconnected nodes in a network. For example, Local Area Networks (LANs) comprise a number of computer based pieces of equipment which are normally distributed within a single establishment. A LAN is most commonly arranged into one of three basic topologies, namely star, bus and ring. More complex network configurations are possible by interconnecting a number of different LANs by means of switches in the form of bridges or routers.

There are a number of different protocols which define the method by which data and commands in the form of messages are transferred from one node to another in a network. In most network protocols, the usual method of routing messages between nodes is to include header information as part of the message which header specifies the address of the destination node and often the address of the node from which the message originates (source address). The addresses specified in the header are those unique addresses which have been assigned to the nodes when the network is configured. In some network schemes, it is necessary to assign a value to a particular node by means of a switch. In a network comprising a number of ring networks interconnected via some form of switch, the address information will commonly have two components identifying the target network and the destination node within the target network.

When a message is sent through a network from a source node to a destination, each node receiving the message processes the destination address information to determine whether the address matches the unique address assigned to that node at power-on. If there is no match, the node forwards the message onto the next (downstream) node which then processes the address information in the same way. In a ring configuration, all the component nodes are generally dual-ported and there is thus no requirement on the nodes to determine on which output path the message should be forwarded. However, in other topologies, in which the component nodes may be single-ported, dual-ported or switch nodes comprising three or more ports, the switch node will contain routing hardware which must determine the identity of the output port for any particular message. In some routing schemes, the switch compares the destination address in the message with a list of addresses contained in a table in order to determine the output node.

There are a number of disadvantages inherent in the prior art routing methods. Firstly, the routing hardware required in each node, especially in a switch node, is complex and also there is the need to assign absolute node addresses at power on.

The present invention seeks to improve on prior addressing schemes and accordingly provides a method of routing a message from a source node to a destination node in a network having a plurality of interconnected nodes, comprising: issuing from the source node a message including a path address value specifying a path between the source node and a destination node, receiving the message at a first node connected to the source node and determining whether a predetermined portion of the path address value corresponds to a predetermined value and if so accepting the message at the first node, and if the predetermined portion of the path address value does not correspond to the predetermined value, modifying the path address value and transferring the message including the modified path address value onto the next node along the path, and continuing said steps of receiving, determining, modifying and transferring at each subsequent node until the path address value corresponds to a predetermined value at which point the message has reached the destination node.

Thus a simple routing method is facilitated, whereby the route address employed in a message allows each node to make a simple routing decision by inspecting a portion of the path address and comparing the portion with a predetermined value. In a preferred routing method, the predetermined value is a zero for all nodes in the network. In this way, the routing hardware in each node is kept very simple.

The type of network in which the method of the present invention finds advantageous use will comprise a mixture of single and dual port nodes which may be arranged in strings, loops or stars and also a number of switch nodes employed to provide an interconnection between the various string or loop portions of the network.

In a preferred method, a dual-ported node on determining that the path address does not correspond to the predetermined value, decrements the portion of the path address before passing onto the downstream node. If the next node is also dual-ported then it will process the path address in the same way. The destination node is thus the node which receives a message including a path address having a zero value. Thus strings or loops of interconnected dual-port nodes are effectively addressed by decrementing a 'hop count'. This is simple, efficient and there is no possibility for a message frame to circulate on a loop indefinitely.

In a preferred method, when a message enters a switch node through one of its three or more ports, the path address component of the message is analysed to determine if that node is the destination node. If not, the switch node determines, from a predetermined portion of the path address, the identity of the output node via which the message is to be forwarded onto the next node. The path address is modified before forwarding by deleting the predetermined portion of the path address. The next node then processes the message including the modified path address.

A further advantage of the routing method of the present invention arises from the modular nature of the path address. Small networks can often be addressed by using a single digit path address. However the Path address can also be expanded to support very large switched networks with 1000's of nodes.

In a preferred routing method, the Path address forms part of a message address field which also includes a Channel address. The Path specifies the route to the destination node and the Channel selects a receiving process within the destination node. One Channel address is advantageously predefined to receive messages and all other Channels are dynamically assigned to inbound data transfers by the destination node. The channel address is modular such that a single 4-bit digit allows up to 15 concurrent inbound transfers. However the address can be expanded to allow 1000's of concurrent inbound transfers for each node.

In a second aspect of the invention, there is provided a method of configuring a network having a plurality of interconnected nodes, comprising: defining one or more initiator nodes within the network; sending a query message from each initiator node to an adjacent node over the path specified by a path address contained in the query message; on receipt of the message, the adjacent node responding with a reply message specifying the number of ports implemented by the adjacent node; sending a configuration message from the initiator node to each node connected to a port implemented on the adjacent node over a path specified in the query message path address; storing the path addresses of each of the nodes connected to the initiator node and the associated number of ports implemented by each node in a configuration table within the initiator node.

Using this configuration technique, the initiator 'walks' through the network one link at a time. Each node addressed by a query message from the initiator sends a reply message including the number of operational ports implemented by the addressed node. The initiator then send a query message to each of the nodes attached to an operational port on the first node. This process continues until the initiator has walked the entire network. The configuration technique according to the invention is thus able to configure a network without the need to set address switches manually as is the case with the prior art SCSI bus.

In a preferred configuration technique, the initiator should be able to readily detect when it has walked back to a node previously visited (which will happen in a cyclic network). This is achieved by assigning a Unique ID to each switch node in the network. When a switch node is addressed by a query message, it returns the Unique ID as well as the number of operational ports in the reply message. The initiator stores the Unique ID value with the corresponding path address and when the same switch node responds to a later query message, the initiator compares the Unique ID with those already contained in the table and is thus able to determine that it has walked round a cyclic network. In a preferred configuration method, each initiator is also assigned a Unique ID.

Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings in which:

Figure 1 depicts the major functional components of a dual-ported node such as may be found in a network using the technique of the present invention;

Figure 2 shows a network comprising a pair of interconnected single-ported nodes;

Figure 3 shows a network configuration comprising a string of interconnected single port and dual port nodes;

Figure 4 shows a network comprising a number of dual-ported nodes interconnected in a loop configuration;

Figure 5 shows a example network of interconnected nodes, including single port, dual port and switch nodes;

Figure 6A shows the format of a frame used in communication between nodes:

Figure 6B shows the format of the address field component of Figure 6A;

Figure 7 shows a network employing one embodiment (scheme A) of the addressing scheme of the present invention;

Figure 8 shows a network employing an alternative embodiment (scheme B) of the addressing scheme of the present invention;

Figures 9A and 9B show the two formats of the Query_node message employed in the network configuration process;

Figure 10 shows the format of the Query_node_reply message employed in the network configuration process;

Figure 11 shows a personal computer network in which the present invention can advantageously

be employed;

Figure 12 shows an example of a file server network in which the present invention may be employed;

Figure 13 shows the major functional components of an N-port node.

The following conventions are used throughout this description:

The bits in an uncoded data byte are numbered 7 to 0 from left to right and Bit 7 is the most-significant bit. The most-significant byte of an integer is first. Bit values are represented as, eg. 1b and Hexadecimal values are represented as, eg. A2h.

The addressing scheme employed in the present invention distinguishes three types of node, according to the connectivity. These are single-port node, dual-port node and switch (3 - 16 ports). In a network employing the present invention, these nodes will typically be electronic devices e.g. computers, printers, storage devices etc.

Figure 1 shows dual-port node 10 including two ports 16, 18 each connected to a serial link 12, 14. Also included is a 3 way router 20 which connects the ports to the node function 22. Depending on the address field, the router forwards an inbound frame to the node itself or to the outbound line of another port. When the node wants to originate a frame it instructs the router to transmit it on a specified port. (All message and data frames relating to a particular command use the same port.)

Figure 13 shows more detail of the major functional components of an N-port node. Each port includes port logic 400;420 including inbound and outbound portions of the serial link. In port 1, the inbound side includes deserialiser and decoder logic 410 which converts the data received over the inbound serial link from bit-serial to parallel data (e.g. 1 byte wide). The deserialiser may also perform clock recovery, bit synchronisation and byte synchronisation. None of these functions are essential to an understanding of the present invention and will not be described further. The incoming data may also need to be decoded e.g. a byte may be represented as a 10 bit character on the link. Logic 410 is connected to Check CRC & FSN logic 414 which checks the CRC information contained with the incoming data and also checks that the Frame Sequence Number (FSN) of the incoming frame corresponds to the expected value. In this way the node is able to check that incoming frames of data are being received in the correct sequence. On the outbound side is Generate CRC & FSN logic 416 and Serialise and encode logic 412, whose function is the corollary of the function of logic 410 and 414 on the inbound side. Port N includes corresponding logic 422, 424, 426 and 428.

Connected to port logic 400 and 428 is router logic 450 which has the function of routing frames of data arriving in the node via one port to either a different port or to a destination process within the node. Included in router logic 450 is Logic 452 which interprets the address field in each frame and updates it according to the rules for 1-port, 2-port or switch nodes as will be described below. This logic generates N request signals, each of which is sent over line 460 to set a latch in a bank of request latches in one of the other ports (e.g. 468 in port N) to request that port to forward the frame. Frame buffer 454 is connected to Interpret & update address logic and is used to temporarily hold a frame. The buffer stores a count of the total number of bytes in the frame, the control field, the updated address field and the data field. A description of the format of a frame including these various fields will be described below. Only a single frame buffer is shown in Figure 13. In practice a port will usually have two or more frame buffers to sustain continuous data transfer. This allows one buffer to be filled from the link while another is being forwarded. Also included in router logic 450 is a multiplexer 456 and a bank of request latches 456. In operation, multiplexer 456 selects a frame which is to be forwarded from the frame buffer of another port e.g. buffer 464 of port N through port 1. The bank of request latches 456 indicates to the multiplexer which frame buffers in other ports contain a frame that is awaiting transmission from port 1. If more than one latch is set simultaneously then a rotating priority algorithm can be used to select a particular request for service.

Router logic 450 contains logic components corresponding to those described for port 1. Logic 472 interprets the address field of a frame and if necessary updates the address. Frame buffer 464 provides temporary storage for a frame. Multiplexer 466 and Request latches 468 control the forwarding of frames over port N. Also included in the router logic are frame buffers 480, multiplexer 482 and request latches 484. These logic components are employed to transfer data which is addressed to this node onto the node function e.g. a destination process within the node. For example this may be a process in a printer which will cause the data to be printed or a process within a disk file which will cause the incoming data to be written to disk. In operation, for a frame received over port 1 which is destined for a process within the node, Interpret and update address logic will generate a request signal which will set a latch in the bank of request latches 484. The set latch informs multiplexer 482 that a frame held in buffer 454 is to be forwarded to the destination process within the node. The frame will the be transferred from buffer 454 to buffer 480.

Turning now to a description of networks in which such a node may be employed, the following are examples of the type of network that can be implemented:

(i) Dedicated connection

Figure 2 shows the simplest case of a dedicated connection between 2 single-port nodes 30 & 32.

(ii) Strings

Figure 3 shows a linear network of dual-port nodes 36,38, 40,42 known as a string. The extreme nodes at either end of a string can be single-port nodes 34, dual-port nodes with one disconnected port or switche nodes.

(iii) Loops

A loop is a cyclic network containing only dual-port nodes 44,45,46,47,48 as shown in Figure 4. A loop provides better availability than a string because any single node can fail without blocking communication between any pair of the remaining nodes. A node can also be inserted into the loop or removed from the loop dynamically without preventing communication between the other nodes.

(iv) Switches

Figure 5 shows a complex network including two switches 106, 114, three strings 100,102,104; 108,110; 116,118 and a cyclic path linking node 118 to switch 114. Switches permit the inter-connection of a large number of nodes. They also allow alternate paths to be provided to achieve fault tolerance.

The two ports at opposite ends of a serial link communicate in units called frames. A frame consists of a sequence of at least 4 data bytes delimited at each end by a special protocol character known as a FLAG. A frame is divided into a sequence of 3 or 4 fields as shown in Figure 6A.

Control field : 1 byte, always present. The control field indicates the frame type and the sequence number. For addressing scheme 'B' it also includes a Digit_delete flag, as described later.

Address field : 1 to 6 bytes, always present. This field is used to route the frame, as described below.

Data field : Up to 128 bytes, optionally present. Except for the configuration messages described later, the contents of the data field are not relevant to an understanding of the present invention.

CRC field : 2 bytes, always present. This field is a standard Cyclic Redundancy Check of the Control, Address and Data fields. It will not be described further.

The maximum lengths of the address and data fields are chosen as a balance between network size, communication efficiency and implementation cost.

The address field of a frame is used firstly to route the frame over a selected Path to the destination node. Path addresses are geographic relative to the source node. This simplifies the routing hardware and it avoids the need to assign absolute node addresses at power-on.

The second function of the address field is to select a Channel within the destination node. A Channel consists of the facilities to receive a message or to receive a single data transfer. Every node must provide a Channel to receive messages. Most nodes also provide at least one Channel to receive data. They may implement additional Channels to support any number of concurrent inbound data transfers. In practice a device may support only 1 data Channel but an adapter or controller will typically provide several.

The upper-level protocol initiates data transfers by exchanging message frames between the source node and the destination node. The destination node allocates a Channel to receive the data frames and it indicates the number of bytes that it can currently accept.

Accordingly the frame address field contains from 1 to 6 bytes, depending on the complexity of the network and the number of Channels implemented by the destination node. The address field begins in the first byte following the control field. It consists of 2 or 3 components and each component is divided into one or more 4-bit digits, as shown in Figure 6B.

Path : This component routes the frame to the destination node. Each digit corresponds to a string or a switch in the path from the source node to the destination node. When a frame is forwarded by a dual-port node the first digit of the path is decremented. When a frame passes through a switch the first 1 or 2 digits of the Path are deleted. This exposes the remainder of the address for subsequent routing decisions. This process is described in much greater detail below.

Channel : This component directs the frame within the destination node. The interpretation depends on the addressing scheme used ('A' or 'B'), as described later. One Channel is predefined to receive messages and the rest are available for inbound data transfers. Typically the data channels will be dynamically allocated by the upper-level protocol.

Pad : If necessary, this is a single digit to make the address field up to an integral number of bytes. The value of the padding digit is normally unimportant since the destination node will have allocated the Channel and it therefore knows how many digits are needed to address it. Each type of node uses different rules to interpret the address field of an inbound frame, as described in the following sections. Two different schemes (A & B)

are described which are functionally equivalent. Scheme 'A' is somewhat simpler to implement in hardware, whereas scheme 'B' tends to have a more compact address field. Both schemes are implemented using logic such as has been described with reference to Figure 13.

SCHEME A

In this scheme each switch deletes the first byte of the Path address when it forwards a frame.

The Channel addressed by the single digit 0h is predefined for all nodes to receive messages. All other Channels are available for data transfer. Thus a single-digit Channel address allows up to 15 data Channels. 2 digits allow up to 240 data Channels, and so on.

In all cases below 'Hi_digit' refers to bits 7:4 of the first byte in the address field and 'Lo-digit' refers to bits 3:0.

The different types of node listed above interpret the address field in a different way. As has been described, Interpret and update address logic 460 or 462 in Figure 13 carries out this process.

Single-port node - this node interprets the address field as follows:

If Hi_digit = 0h then Do;
Accept the frame;
Interpret the remainder of the address field as a Channel;
End;
Else reject the frame;

The port in a single port node is numbered '0' for reference by the Query_node_reply message, details of which are described below.

Dual-port node - this node interprets the address field as follows:

If Hi_digit = 0h then Do;
Accept the frame;
Interpret the remainder of the address field as a Channel;
End;
Else Do;
Decrement Hi_digit;
Forward the frame on the other port;
End;

Dual-port nodes treat the first digit of the address field as a 'hop count' of the nodes to be traversed. The destination node in a string or loop is located by decrementing the hop count until it reaches zero. Using scheme A, in order to allow unrestricted communication between any two nodes the maximum number of nodes which may be formed in a string is 17, including the end nodes. This is due to the use of a single Hex digit to address all nodes in the string. For example, this allows 16 devices to be connected to a single adapter port.

The two ports of a dual port node are numbered '0' and '1' for reference by the Query_node_reply message, details of which are described below.

Switch node - this node interprets the address field as follows:

If Hi_digit = 0h then Do;
If Lo_digit = 0h then Do;
Accept the frame;
Interpret the remainder of the address field as a Channel;
End;
Else Do;
Select Output_port = (Input_port + Lo_digit) Modulo 16;
Delete the first address byte;
Forward the frame via Output_port;
End;
End;
Else reject the frame;

The switch ports are numbered sequentially by the implementation, starting from '0'. (Note that the port numbers are fixed, but frame addressing is relative to the input port.) In practice all initiators must execute the Configuration process to discover the nodes that are operational and their Path addresses. It is not possible to have a switch forward a frame out of the same port that received it.

An example of a network employing scheme A will now be described with reference to Figure 7 which shows a network topology corresponding to that shown in Figure 5. The numbers in the edges of the boxes are the (assumed) port numbers. For illustration purposes only, the ports are numbered clockwise from the port nearest the indicated 'Initiator'. The number inside each box is the hexadecimal Path address of that node relative to that Initiator. The dual-port nodes on the cyclic path 216, 218 have two addresses depending on the Path used by the Initiator.

Taking as an example the routing of a message from initiator node 200 to dual-port node 210, the path address specified in the address field will comprise 221h. Node 202 on receiving the message determines in accordance with the above defined dual-port algorithm that Hi_digit is not equal to 0h and accordingly decrements Hi_digit from 2h to 1h. Similarly, dual port node 204 decrements Hi_digit 1h to 0h and forwards the message on to switch 206. The switch examines both Hi_digit (0h) and Lo_digit (2h), determines that both Hi and Lo are not equal to 0h and then in accordance with the Switch algorithm defined above determines on which of its ports the message should be forwarded. The input port is numbered 0 and thus (Input_port + Lo_digit) Modulo 16 is determined to be equal to 2. Before forwarding the message from port 2, the switch deletes the

first address byte to leave a path address of 1h. Node 208 examines the address and decrements the path address value to 0h. Node 210 in turn processes the path address, determines that it is the destination node and accordingly accepts the message.

Scheme B

In this scheme each switch deletes the first 1 or 2 digits of the Path address. This is accomplished by defining a Digit_delete flag in the frame Control field. The first digit is logically deleted by setting Digit_delete. The second digit is deleted by clearing Digit_delete and physically deleting the first byte of the address field. The third digit is then logically deleted by setting Digit_delete, and so on.

All Channels beginning with the digit 0h are invalid. The Channel addressed by the single digit 1h is predefined for all nodes to receive messages. All other Channels are available for data transfer. Thus a single-digit Channel address allows up to 14 data Channels. 2 digits allow up to 224 data Channels, and so on. In all cases below 'Next_digit' refers to the next active digit in the current address field, after taking account of Digit_delete in the control field and digits that have been deleted by previous rules.

Single-port node - this node interprets the address field as follows:
Accept the frame;
If Next_digit = 0h then delete Next_digit;
If (new) Next_digit = 0h then delete Next_digit;
Interpret the remainder of the address field as a Channel;

One leading zero needs to be discarded when a single-port node is at the end of a string. Two leading zero's need to be discarded when a single-port node receives a Query_node message during the configuration process.

Dual-port node - this node interprets the address field as follows:
If Next_digit = 0h then Do;
Accept the frame;
Discard Next_digit;
If (new) Next_digit = 0h then discard Next_digit;
Interpret the remainder of the address field as a Channel;
End;
Else Do;
Decrement Next_digit;
Forward the frame on the other port;
End;

One zero needs to be deleted before the Channel address when a dual-port node receives a Query_node message during the configuration process.

Switch node - this type of node interprets the address field as follows:
If Next_digit = 0h then discard Next_digit;
If (new) Next-digit = 0h then Do;
Accept the frame;
Delete Next_digit;
Interpret the remainder of the address field as a Channel;
End;
Else Do;
Select Output_port = (Input_port + (new) Next_digit) Modulo 16;
Delete (new) Next_digit;
Forward the frame via the selected port;
End;

One leading zero needs to be deleted when a switch is at the end of a string.

An example of a network employing addressing scheme B will now be described with reference to Figure 8. Taking the routing of a message from node 300 to 310 as an example, the path address in the message frame address field will be 221h. Node 302 decrements Next_digit to 1h and forwards the message including a path address of 121h onto node 304. At node 304, Next_digit is decremented to 0h and the message including a path address of 021 on to switch node 306. Node 306 discards Next_digit and calculates the output port as port 2. The (new) Next_digit is discarded and the message including a path address of 1h is forwarded to node 308, which in turn decrements Next_digit to 0h. Node 310 on receipt of the message determines that it is the destination node, accepts the message and determines the channel address within the node to which the message is directed.

It will be noted that with scheme B nodes 314, 316 and 318 are addressable using a 4-digit path address in contrast to the 5-digit path address of the corresponding nodes 214, 216 and 218 in Figure 7. Thus it will usually be the case that a smaller number of digits will be necessary to address nodes in a network employing scheme B.

Next will be described the technique employed for configuring the network. For each command sent over the network, a node can be classified as either (i) an Initiator, ie. the node that issued the command, or (ii) a Target, ie. the node that received the command.

Each potential Initiator must perform a configuration process to determine the other nodes that are present in the network and their path address(es). The transport layer defines the Query_node and Query_node_reply to support configuration. The upper-level protocol is expected to define commands to retrieve Vital Product Data such as the device type and serial number. For example,

SCSI provides the Inquiry command.

All switches and initiators must be assigned a Unique_ID when the node is manufactured. The Unique_ID is typically stored in EPROM and consists of a 4-byte vendor identification followed by a 4-byte node identification assigned by the manufacturer. The IDs for both switches and initiators are unsigned binary integers. The Unique_ID is used during the configuration process to detect cyclic networks.

In the configuration process each Initiator 'walks' the network one link at a time. First the Initiator chooses one of its operational ports and issues a Query_node message to the adjacent node. The adjacent node returns a Query_node_reply to indicate how many ports it has, which ports are operational and its Unique_ID, if any. The Initiator enters this information into a Configuration table together with the corresponding Path address. (Typically the Configuration table will also contain other information provided by the upper-level protocol, eg. the device type and serial number from the SCSI Inquiry command.) The Initiator then issues a Query_node to the next node via one of the operational ports on the adjacent node and so on until it has walked the entire network.

If the network is cyclic then an Initiator must detect when it has walked back to a node previously visited. This can be achieved by comparing the Unique_ID received from switch nodes and other initiators with previous entries in the Configuration table.

If a node has a port which is not operational then an Initiator does not attempt to walk that link during configuration. (This would produce an error.) If the port subsequently becomes operational each Initiator will be alerted by an asynchronous message from the corresponding node. Then each Initiator walks the new link and adds nodes to its Configuration table until it encounters a single-port node, another port which is not operational or it returns to a Unique_ID that is already known.

Each Initiator must perform the full Configuration process when it powers on. It must also perform a partial configuration when it receives an asynchronous message as a result of a new link becoming operational.

If a link encounters a permanent error (eg. the link has been disconnected) each Initiator is alerted by an asynchronous message from the nearer of the 2 nodes connected by that link. Each Initiator must then delete the path(s) to those nodes beyond the error from its Configuration table.

A node that is just a Target (eg. a device) does not need to perform the Configuration process or build a Configuration table. It just responds to the Query_node messages from the Initiators.

Two types of message are defined to support the configuration process, namely Query_node and Query_node_reply.

A Query_node message is sent from an Initiator to every other operational node during the configuration process. The destination node returns a Query_node_reply. This exchange also provides a remote wrap test to verify the integrity of the path.

On entry to the destination node the value remaining in the address field of the Query_node message should be 0000h for scheme 'A'. For scheme 'B', after taking account of the Digit_delete flag, the remaining address field should be 001h plus a possible pad digit. This guarantees that the message will be accepted regardless of whether the node is a single-port, dual-port or switch node. If an Initiator intends to use several alternate paths to the same Target node then it must issue Query_node once over each path.

The Query_node message has two slightly different formats which are shown in Figure 10. Format 1 is perceived only in scheme A when a single-port or dual-port node receives a Query_node message. In all other cases format 2 is used. The components of the two formats are as follows:

Message_code : identifies the message as 'Query_node'.

Pad : this byte arises because in scheme 'A' a single-port or dual-port node will interpret the last byte of the address field as the Message_code.

Tag : this 2-byte field is returned in the Query_node_reply message. The Tag is assigned by the Initiator and it must be unique among the Tags that are currently active from that Initiator.

Return_address : this 4-byte field specifies the value that should be placed in the address field of the resulting Query_node_reply message. It contains the whole address field, including the Path and the Channel, left-aligned and padded to 4 bytes. For scheme 'A' the Channel is 0h and the padding is zero or more digits of Fh. For scheme 'B' the Channel is 1h and the padding is zero or more digits of 0h. This allows the destination node to determine the significant digits.

Unique_ID : this 8-byte field contains the Unique_ID of the Initiator that issued the Query_node message.

The second type of message is the Query_node_reply which every node must return in response to a Query_node message. Query_node_reply is returned on the same port that received the corresponding Query_node message. It indicates the total number of ports implemented by the addressed node, which ports are operational and the port currently being used. A port is operational when it is receiving a signal from the re-

mote node. The components of the Query_node_reply message are shown in Figure 11.

Message_code : this byte identifies the message as 'Query_node_reply'. Current_port Bits 7:4 contain an unsigned binary integer that indicates which port is currently being used. Total_ports Bits 3:0 contains an unsigned binary integer that is one less than the number of ports implemented.

Tag : this 2-byte field is copied from the Query_node message. It identifies the Query_node message for which this reply is being generated.

Port_mask : this 16-bit field indicates which ports are currently operational. The left-most bit is set if port 0 is operational and so on.

ULP : this byte identifies the upper-level protocol to communicate with the node. The only value currently defined is '00'X indicating 'SCSI-2'.

Init : when set to 1b, bit 0 indicates that the node is an Initiator.

Unique_ID : this 8-byte field is only present if the node is an Initiator or a switch.

The address field in a Query_node_reply frame is copied from the Return_address field in the corresponding Query_node message. However any complete padding bytes must be discarded.

The network addressing and configuration schemes described above may be used in a variety of different applications, two examples of which are now described. It will be appreciated that the present invention can readily be used in other types of network.

Personal Computer : A string of dual-ported devices is particularly attractive for connecting I/O devices to a personal computer, as shown in Figure 11. Adapter 50 which will typically reside in the system unit of a personal computer is attached via link 51 to disk drive 52 which is in turn attached to disk drive 54 via link 53 which is in turn attached to printer 56 via link 55. The use of a string reduces the attachment cost per device and it avoids wiring congestion at the adapter. Using the method and apparatus of the present invention, the routing hardware in each of the two disk drives and printer is kept simple. Furthermore, there is no need for these devices to have addresses already assigned which with some devices avoids the need to manually assign addresses by means of a switch when the device is added to the string. Optionally the loop can be closed by provision of link 57 to provide increased bandwidth or a measure of fault-tolerance.

File server : High availability is important in a shared system such as a file server. This application also requires dual-ported disk drives, but this time the main reason for the second port is to provide a backup path in the event of a failure in the primary attachment path. Therefore in practice all serial disk drives will probably be dual-ported. In conjunction with disk arrays, dual-ported disk drives allow configurations with no single point of failure, as shown in Figure 12. In this configuration a pair of servers 60 and 62 are connected via dedicated links to both switches 64 and each port of the dual-ported disk drives 68, 70 72 and 74 is connected to one of the switches. The use of dedicated links to each drive allow full concurrent maintenance with no impact to the operation of other drives.

Claims

1. A method of routing a message from a source node to a destination node in a data processing system configured as a network having a plurality of interconnected nodes, comprising:

   issuing from the source node a message including a path address value specifying a path between the source node and a destination node,

   receiving the message at a first node connected to the source node and determining whether a predetermined portion of the path address value corresponds to a predetermined value and if so accepting the message at the first node;

   and if the predetermined portion of the path address value does not correspond to the predetermined value, modifying the path address value and transferring the message including the modified path address value onto the next node along the path,

   and continuing said steps of receiving, determining, modifying and transferring at each subsequent node until the path address value corresponds to a predetermined value at which point the message has reached the destination node.

2. A method as claimed in claim 1, the network including dual-port nodes, wherein if on receipt of a message from a preceding node, the dual-port node determines that the predetermined portion of path address does not correspond to the predetermined value for a dual-port node, the dual-port node decrements the value of the predetermined portion of the path address before forwarding the message onto the next node.

3. A method as claimed in claim 2, wherein the predetermined value for a dual-ported node is zero.

4. A method as claimed in any preceding claim, the network including one or more switch

nodes, each having three or more ports, wherein if on receipt of a message from a preceding node the switch node determines that the predetermined portion of path address does not correspond to the predetermined value for a switch node, the switch node calculates from the path address the identity of the port through which the message should be passed and deletes a portion of the path address before transferring to the node connected to the determined port.

5. A method as claimed in any preceding claim, wherein the path address value forms part of a message address field, the address field further including a channel address which specifies a receiving process within the destination node to which the message should be directed.

6. A method of configuring a network having a plurality of interconnected nodes, comprising:

defining one or more initiator nodes within the network;

sending a query message from each initiator node to an adjacent node over the path specified by a path address contained in the query message;

on receipt of the message, the adjacent node responding with a reply message specifying the number of ports implemented by the adjacent node;

sending a configuration message from the initiator node to each node connected to a port implemented on the adjacent node over a path specified in the query message path address;

storing the path addresses of each of the nodes connected to the initiator node and the associated number of ports implemented by each node in a configuration table within the initiator node.

7. Message routing apparatus for use in a node being one component of a network of interconnected nodes, comprising:

means for receiving a message from an adjacent node, the message including a path address value specifying the route through the network which the message is to take to a destination node;

means for determining whether a predetermined portion of the path address value corresponds to a predetermined value and if so accepting the message; and

means for modifying the path address value if the predetermined portion does not correspond to the the predetermined value and for forwarding the message including the modified path address value onto the next node.

8. Message routing apparatus as claimed in claim 7, for use in a multi-port switch node within a network, the means for modifying comprising means for calculating from the path address the identity of the port through which the message should be transferred and for deleting a portion of the path address before forwarding the message onto the node connected to the determined port.

9. A network comprising a plurality of dual-ported nodes, connected to at least one predefined initiator node, each dual port node including message routing apparatus as claimed in claim 7.

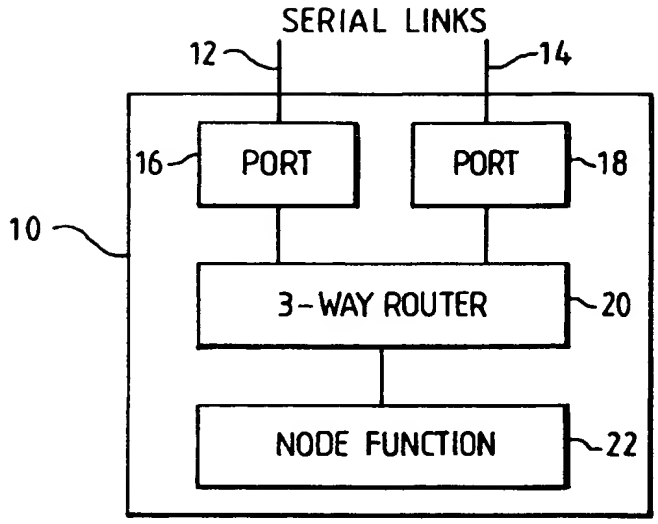

FIG. 1

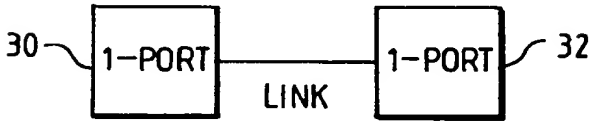

FIG. 2

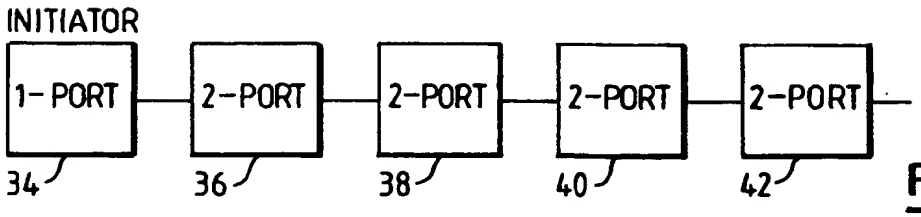

FIG. 3

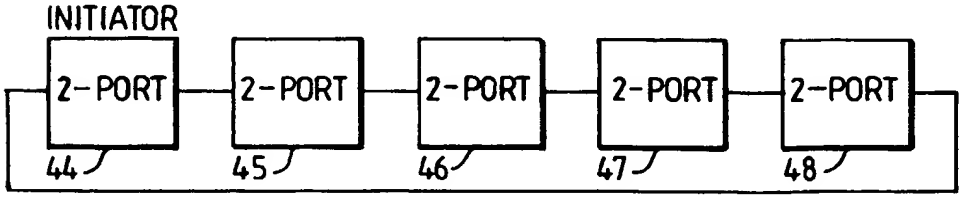

FIG. 4

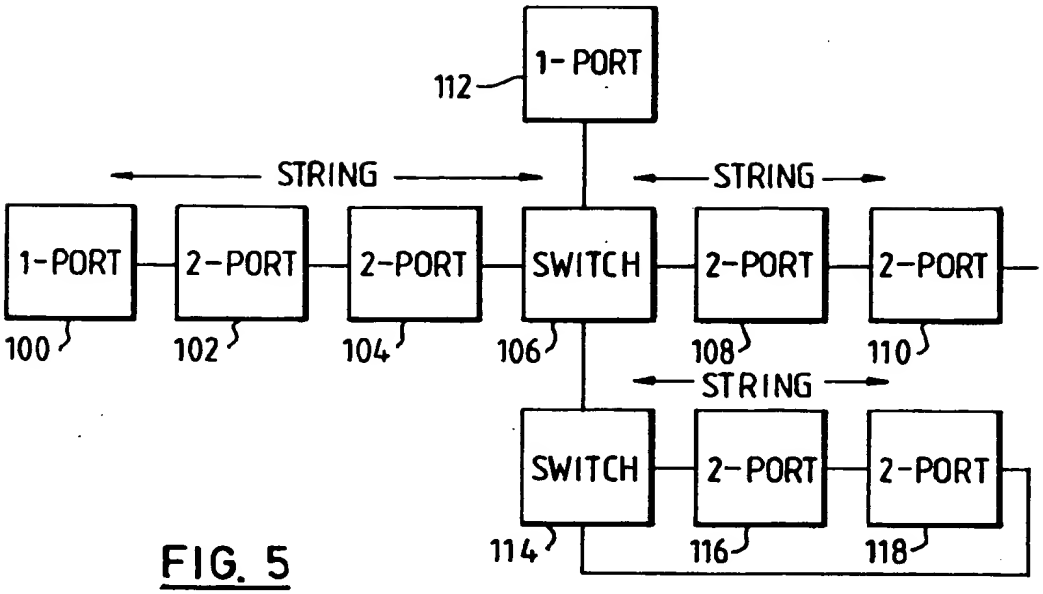

FIG. 5

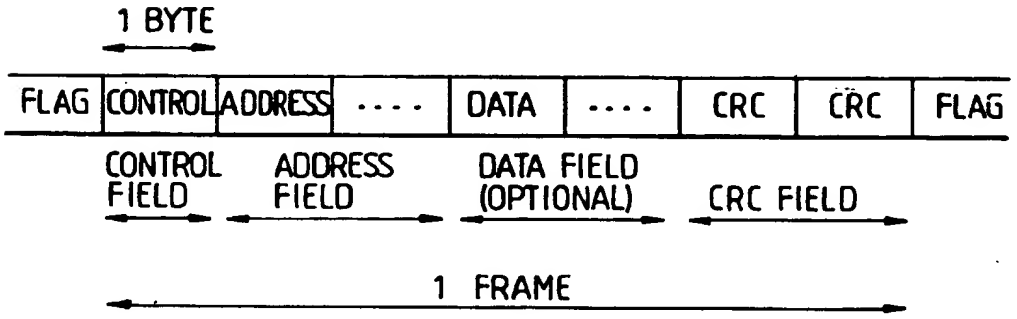

FIG. 6A

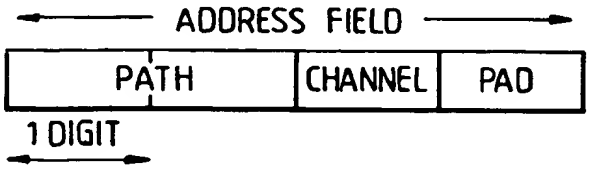

FIG. 6B

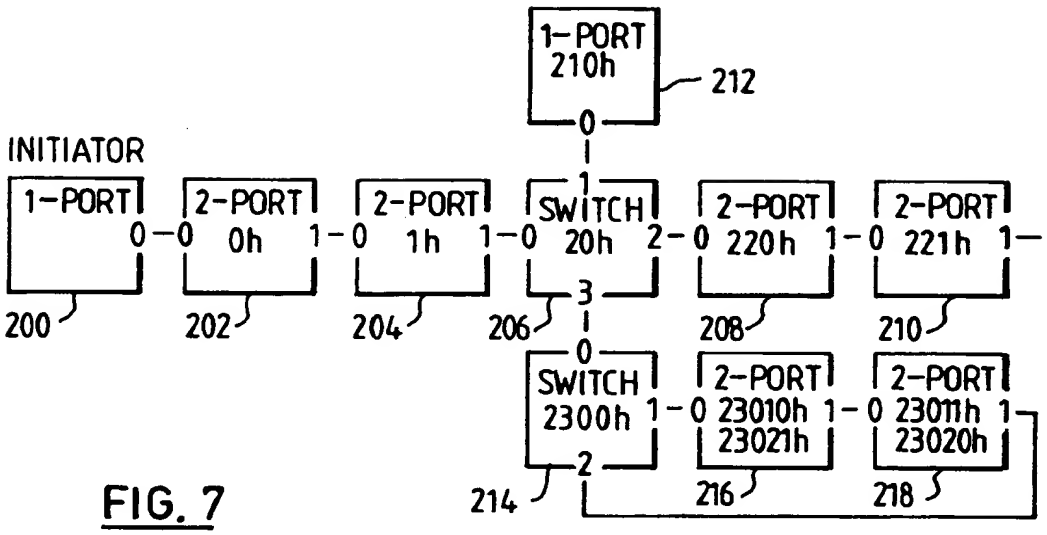
1-PORT
210h
212
INITIATOR
1-PORT
2-PORT
0h
2-PORT
1h
SWITCH
20h
2-PORT
220h
2-PORT
221h
200
202
204
206
208
210
SWITCH
2300h
2-PORT
23010h
23021h
2-PORT
23011h
23020h
214
216
218

FIG. 7

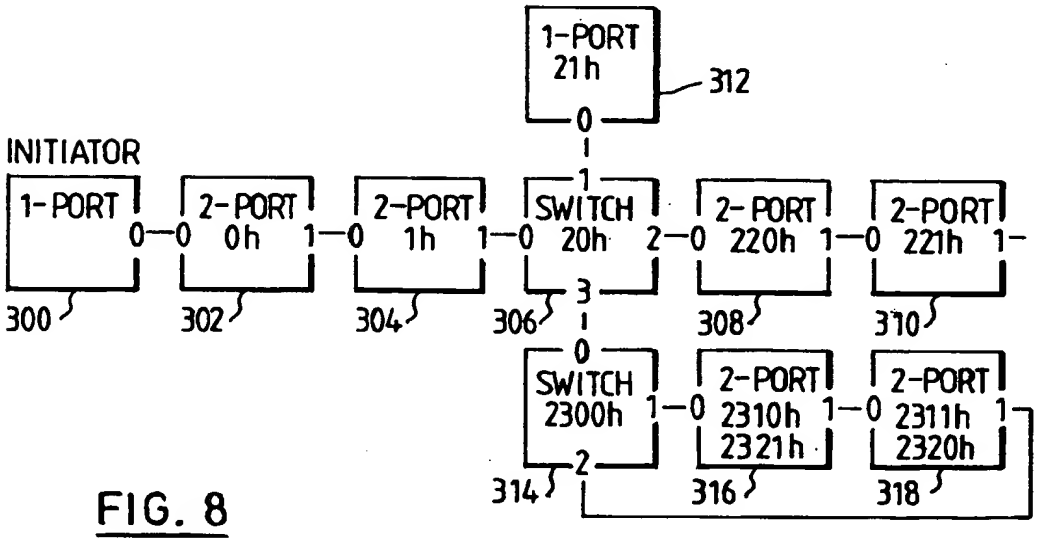
1-PORT
21h
312
INITIATOR
1-PORT
2-PORT
0h
2-PORT
1h
SWITCH
20h
2-PORT
220h
2-PORT
221h
300
302
304
306
308
310
SWITCH
2300h
2-PORT
2310h
2321h
2-PORT
2311h
2320h
314
316
318

FIG. 8

FORMAT 1 :

BYTE 0	MESSAGE_CODE = 00h
1	PAD 00h
2	TAG
3	
4	RETURN_ADDRESS
.	
7	
8	UNIQUE_ID
.	
15	

FORMAT 2 :

BYTE 0	MESSAGE_CODE = 00h
1	TAG
2	
3	RETURN_ADDRESS
.	
6	
7	UNIQUE_ID
.	
14	

FIG. 9

BYTE 0	MESSAGE_CODE = 01h	
1	CURRENT_PORT	TOTAL_PORTS
2	TAG	
3		
4	PORT_MASK	
5		
6	UPPER_LEVEL_PROTOCOL (ULP)	
7	0 0 0 0 0 0 0	INIT
8		
.	UNIQUE_ID	
15		

FIG.10

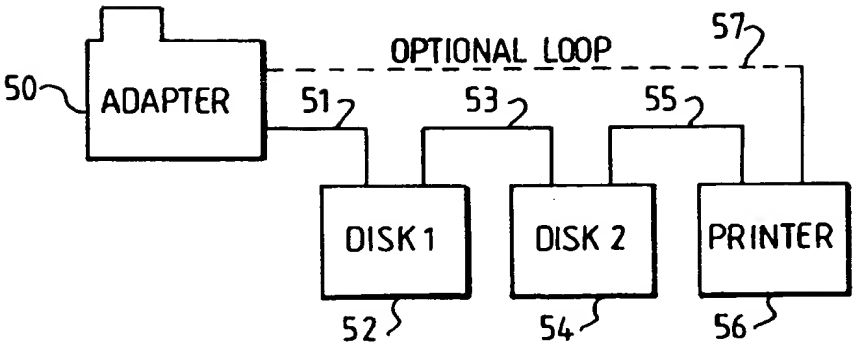
OPTIONAL LOOP
57
50
ADAPTER
51
53
55
DISK 1
DISK 2
PRINTER
52
54
56

FIG. 11

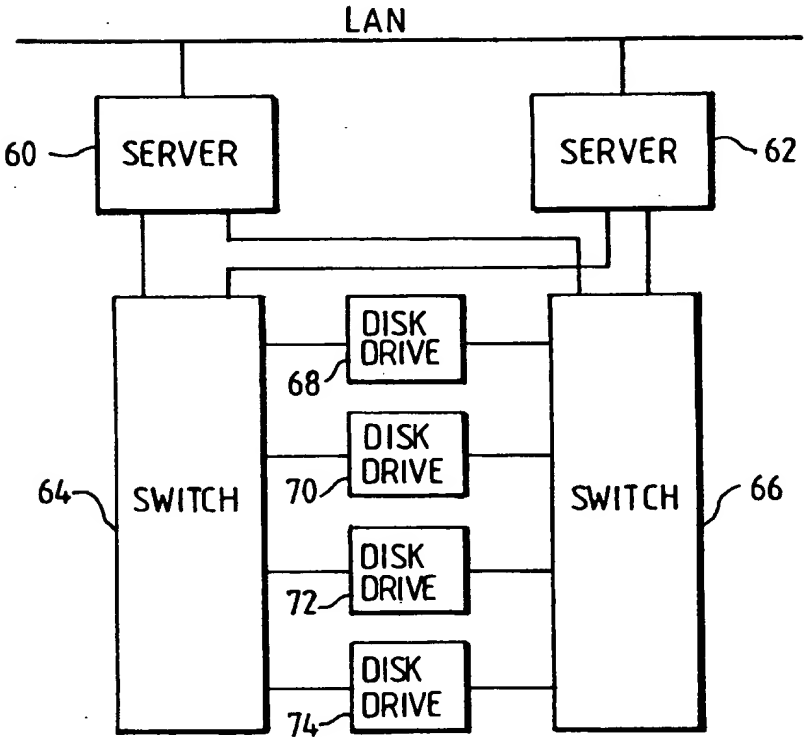
LAN
60
SERVER
SERVER
62
DISK DRIVE
68
DISK DRIVE
70
DISK DRIVE
72
DISK DRIVE
74
64
SWITCH
SWITCH
66

FIG. 12

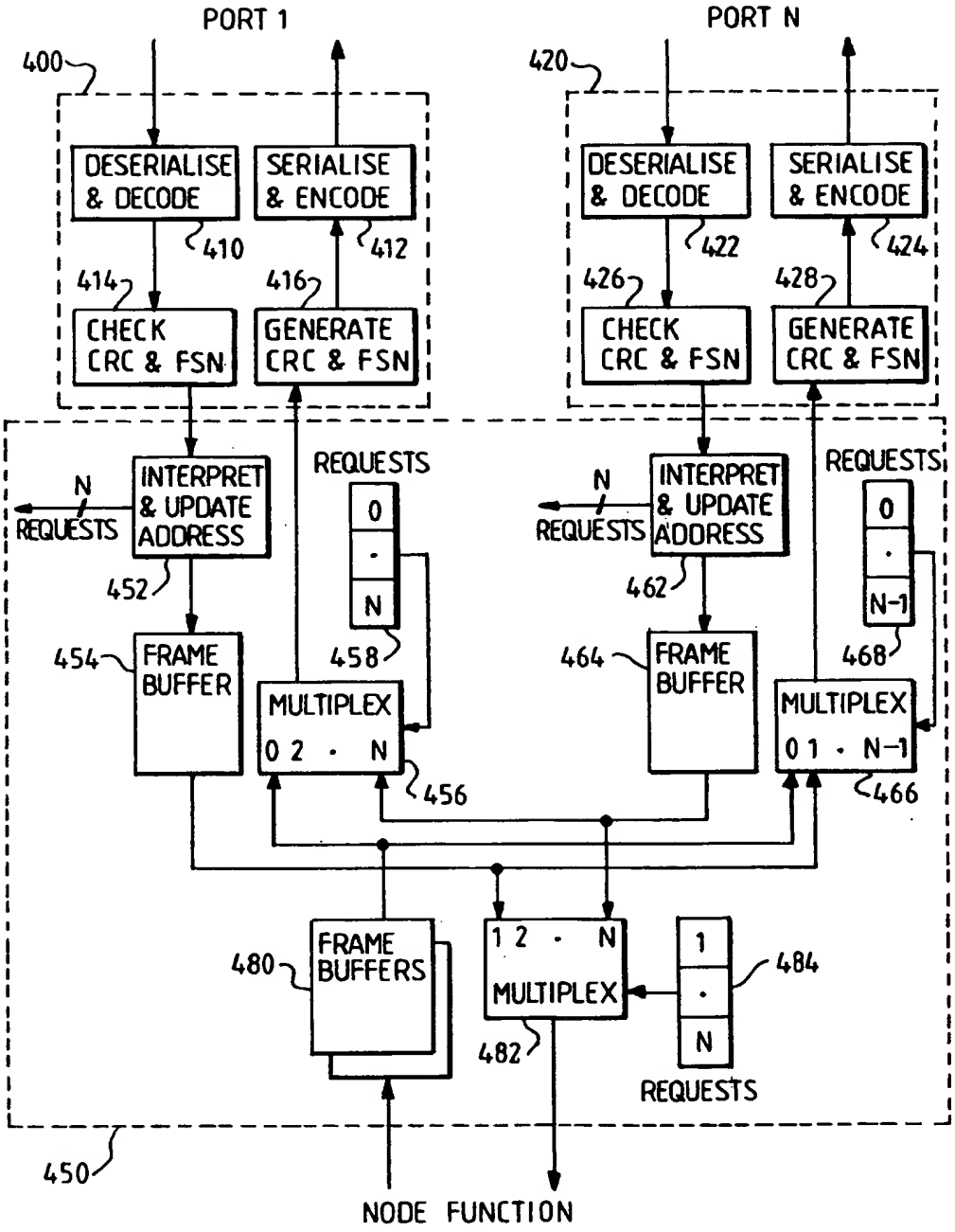
PORT 1
PORT N
400
420
DESERIALISE & DECODE
SERIALISE & ENCODE
410
412
422
424
414
416
426
428
CHECK CRC & FSN
GENERATE CRC & FSN
INTERPRET & UPDATE ADDRESS
N REQUESTS
REQUESTS
0
.
N
N-1
452
462
454
464
458
468
FRAME BUFFER
MULTIPLEX
0 2 · N
0 1 · N-1
456
466
480
FRAME BUFFERS
1 2 · N
1
484
482
450
NODE FUNCTION

FIG.13